REMARKS

Claims 1, 2, and 8-29 are pending in the present application. Claims 3-7 have been cancelled without prejudice or disclaimer to the subject matter contained therein.

Rejection under 35 U.S.C. §102(e)

Claims 1-29 have been rejected under 35 U.S.C. §102(e) for being anticipated by Seseck et al. (Published US Patent Application 2003/0086098). This rejection is respectfully traversed.

Independent Claim 1

In formulating the rejection of independent claim 1 under 35 U.S.C. §102(e), the Examiner alleges that Seseck et al. discloses, at paragraph **[0030]**, a method for creating reusable composite components from interpreted pages of rendered document during dynamic document construction. Moreover, the Examiner alleges that Seseck et al. discloses, at paragraphs **[0031]** and **[0049]**, obtaining a list of document components from the page and identifying any non-cached components; at paragraphs **[0030]** and **[0032]**, caching individual reusable document components rendered to their respective bounding box dimensions; at paragraph **[0032]**, permuting the reusable document components into composite combinations of reusable document components; at paragraph **[0032]**, caching each of composite reusable document component rendered relative to each other in a bounding box of sufficient size to adequately contain the combination; at paragraph **[0034]**, combining reusable document components in their relative positions to form composite reusable underlays; and, at paragraph **[0034]**, caching said composite reusable underlays rendered to full page size. Based upon these allegations, the Examiner concludes that the presently claimed invention is anticipated by Seseck et al. These allegations and conclusion are respectfully traversed.

As set forth above, independent claim 1 recites a method for creating reusable composite components from interpreted pages of rendered document during dynamic document construction by obtaining a list of document components from the page and identifying any non-cached components; caching individual reusable document components rendered to their respective bounding box dimensions; permuting the reusable document components into composite combinations of reusable document components; caching each of composite reusable document component rendered relative to each other in a bounding box of sufficient size to adequately contain the combination; combining reusable document components in their relative positions to form composite reusable underlays; and caching said composite reusable underlays rendered to full page size.

For example, independent claim 1 explicitly recites caching individual reusable document components rendered to their respective bounding box dimensions. In contrast, Seseck et al. teaches caching complete rendered documents.

More specifically, Seseck et al. discloses, at paragraph **[0030]**:

> The process carried out by a typical printer in printing a document involves receiving a print request and a document to be printed from the requester. The document is converted to a particular page description language (PDL) supported by the printer (e.g., a particular PCL (Printer Control Language) level, Postscript, etc.), in a process referred to as "rendering." This rendering is typically performed at the computing device submitting the print requested. The rendered document is then converted to a set of hardware-ready bits, in a process referred to as "ripping." This ripping is typically performed at the printer. These hardware-ready bits are supplied directly to the print engine, which controls the application of a print substance (e.g., toner or ink) onto a print medium (e.g., paper, cloth, transparency, etc.) either directly or indirectly (e.g. via a transfer belt) so that the print substance is applied in accordance with the ripped data. This printing process is well-known to those skilled in the art.

At paragraph **[0032]**, Seseck et al. discloses:

> Pre-printing controller 156 maintains a record of print-ready documents in a print-ready document list 164. The print-ready documents are those which have been previously received by printer 132 and converted into a print-ready form. The print-ready documents themselves are stored in a print-ready file cache 166 in mass storage device 152. Alternatively, cache 166 may be stored in memory (e.g., memory 150), or stored on a separate device (e.g. a network drive) remote from printer 132 (not shown).

As disclosed by Seseck et al., paragraphs **[0030]** and **[0032]** of Seseck et al. teach that print-ready documents are stored in a print-ready file cache in a mass storage device. Paragraphs **[0030]** and **[0032]** of Seseck et al. fail to disclose caching individual reusable document **components**. Moreover, paragraphs **[0030]** and **[0032]** of Seseck et al. fail to disclose caching individual reusable document **components rendered to their respective bounding box dimensions** because Seseck et al. teaches caching whole documents, not components thereof nor components rendered to respective bounding box dimensions.

Furthermore, independent claim 1 explicitly recites permuting the reusable document components into composite combinations of reusable document components. In contrast, Seseck et al. teaches caching complete rendered documents.

At paragraph **[0032]**, Seseck et al. discloses:

> Pre-printing controller 156 maintains a record of print-ready documents in a print-ready document list 164. The print-ready documents are those which have been previously received by printer 132 and converted into a print-ready form. The print-ready documents themselves are stored in a print-ready file cache 166 in mass storage device 152. Alternatively, cache 166 may be stored in memory (e.g., memory 150), or stored on a separate device (e.g. a network drive) remote from printer 132 (not shown).

As disclosed by Seseck et al., paragraph **[0032]** of Seseck et al. teaches that print-ready documents are stored in a print-ready file cache in a mass storage device. Paragraph **[0032]** of Seseck et al. fails to disclose **permuting** the reusable document components. Moreover, paragraph **[0032]** of Seseck et al. fails to disclose **permuting** the reusable document components **into composite combinations of reusable document components** because Seseck et al. teaches caching whole documents, not permuting reusable document components nor permuting reusable document components into composite combinations of reusable document components.

Also, independent claim 1 explicitly recites caching each of composite reusable document component rendered relative to each other in a bounding box of sufficient size to adequately contain the combination. In contrast, Seseck et al. teaches caching complete rendered documents.

At paragraph **[0032]**, Seseck et al. discloses:

> Pre-printing controller 156 maintains a record of print-ready documents in a print-ready document list 164. The print-ready documents are those which have been previously received by printer 132 and converted into a print-ready form. The print-ready documents themselves are stored in a print-ready file cache 166 in mass storage device 152.

> Alternatively, cache 166 may be stored in memory (e.g., memory 150), or stored on a separate device (e.g. a network drive) remote from printer 132 (not shown).

As disclosed by Seseck et al., paragraph **[0032]** of Seseck et al. teaches that print-ready documents are stored in a print-ready file cache in a mass storage device. Paragraph **[0032]** of Seseck et al. fails to disclose **caching each of composite reusable document component** rendered relative to each other **in a bounding box of sufficient size to adequately contain the combination** because Seseck et al. teaches caching whole documents, not caching each of composite reusable document component rendered relative to each other in a bounding box of sufficient size to adequately contain the combination.

Independent claim 1 further explicitly recites combining reusable document components in their relative positions to form composite reusable underlays. In contrast, Seseck et al. teaches comparing web pages to cached pages.

At paragraph **[0034]**, Seseck et al. discloses:

> The web page data is received by browsing module 158 and stored in memory 150. Some of the data received as part of a web page may include a timestamp indicating the last time the web page was modified (the timestamp may be part of the page that is displayed by a browser when displaying the page, or alternatively included in control information that is not displayed). Controller 156 compares the web page to print-ready document list 164 to determine whether a print-ready version of this web page is already stored in cache 166. This comparison can be performed in a variety of different manners, and in one implementation list 164 includes, for each file in cache 166, the URL of the web page corresponding to the file. The URL received from client 134 can then be compared to the URLs in list 164, and if the received URL exists in list 164 then a print-ready version of the document is already in cache 166. Alternatively, the determination of whether a print-ready version of a web page is already stored in cache 166 may be performed after the indication of the web page is received from client 134, but prior to retrieval of the web page from server 136 by browsing module 158. However, even if the web page does already exist in cache 166, at least a portion of the web page is still retrieved from server 136 so that the timestamps can be compared, as discussed below.

As disclosed by Seseck et al., paragraph **[0034]** of Seseck et al. teaches that a controller compares a web page to a print-ready document list to determine whether a print-ready version of the web page is already stored in cache. Paragraph **[0034]** of Seseck et al. fails to disclose **combining reusable document components in their relative positions to form composite reusable underlays** because Seseck et al. teaches the comparing of a web page to a print-ready document list to determine whether a print-ready version of the web page is already stored in cache, not combining reusable document components in their relative positions to form composite reusable underlays.

In summary, Seseck et al. fails to disclose caching individual reusable document components rendered to their respective bounding box dimensions; permuting the reusable document components into composite combinations of reusable document components; caching each of composite reusable document component rendered relative to each other in a bounding box of sufficient size to adequately contain the combination; and/or combining reusable document components in their relative positions to form composite reusable underlays.

Therefore, contrary to the Examiner's allegations, Seseck et al. fails to anticipate the presently claimed invention as set by independent claim 1.

Independent Claim 2

In formulating the rejection of independent claim 2 under 35 U.S.C. §102(e), the Examiner alleges that Seseck et al. discloses, at paragraph **[0030]**, a method for rendering pages having a combination of reusable components and non-cached components. Moreover, the Examiner alleges that Seseck et al. discloses, at paragraph **[0030]**, assessing said rendered page for the possibility of having an underlay-overlay pair; at paragraphs **[0032]** and **[0034]**, searching a cache of reusable underlays for underlays having the reusable document components needed by the page; at paragraph **[0036]**, if the correct reusable underlay is not found in cache then generating a composite reusable underlay from the reusable document components of said page and caching said reusable underlay rendered to full page size; at paragraph **[0036]**, creating a full page size overlay from the non-cached components that is retained until it is mated with the cached reusable underlay; at paragraph **[0036]**, if the correct underlay is found in cache then retrieving the reusable underlay; and, at paragraph **[0037]**, rendering, along with the overlay, the page therefrom. Based upon these allegations, the Examiner concludes that the presently claimed invention is anticipated by Seseck et al. These allegations and conclusion are respectfully traversed.

As set forth above, independent claim 2 recites a method for rendering pages having a combination of reusable components and non-cached components by assessing said rendered page for the possibility of having an underlay-overlay pair; searching a cache of reusable underlays for underlays having the reusable document components needed by the page; if the correct reusable underlay is not found in cache then generating a composite reusable underlay from the reusable document components of said page and caching said reusable underlay rendered to full page size; creating a full page size overlay from the non-cached components that is retained until it is mated with the cached reusable underlay; if the correct underlay is found in cache then retrieving the reusable underlay; and, rendering, along with the overlay, the page therefrom.

For example, independent claim 2 explicitly recites searching a cache of reusable underlays for underlays having the reusable document components needed by the page. In contrast, Seseck et al. teaches storing print-ready documents.

More specifically, Seseck et al. discloses, at paragraph **[0032]**:

> Pre-printing controller 156 maintains a record of print-ready documents in a print-ready document list 164. The print-ready documents are those which have been previously received by printer 132 and converted into a print-ready form. The print-ready documents themselves are stored in a print-ready file cache 166 in mass storage device 152. Alternatively, cache 166 may be stored in memory (e.g., memory 150), or stored on a separate device (e.g. a network drive) remote from printer 132 (not shown).

As disclosed by Seseck et al., paragraph **[0032]** of Seseck et al. discloses that the print-ready documents are stored in a print-ready cache; i.e., complete documents are stored in the cache. On the other hand, independent claim 2 recites **searching** a cache of reusable underlays **for underlays having the reusable document components needed by the page**.

At paragraph **[0034]**, Seseck et al. discloses:

The web page data is received by browsing module 158 and stored in memory 150. Some of the data received as part of a web page may include a timestamp indicating the last time the web page was modified (the timestamp may be part of the page that is displayed by a browser when displaying the page, or alternatively included in control information that is not displayed). Controller 156 compares the web page to print-ready document list 164 to determine whether a print-ready version of this web page is already stored in cache 166. This comparison can be performed in a variety of different manners, and in one implementation list 164 includes, for each file in cache 166, the URL of the web page corresponding to the file. The URL received from client 134 can then be compared to the URLs in list 164, and if the received URL exists in list 164 then a print-ready version of the document is already in cache 166. Alternatively, the determination of whether a print-ready version of a web page is already stored in cache 166 may be performed after the indication of the web page is received from client 134, but prior to retrieval of the web page from server 136 by browsing module 158. However, even if the web page does already exist in cache 166, at least a portion of the web page is still retrieved from server 136 so that the timestamps can be compared, as discussed below.

As disclosed by Seseck et al., paragraph **[0034]** of Seseck et al. teaches that a controller compares a web page to a print-ready document list to determine whether a print-ready version of the web page is already stored in cache. Paragraph **[0034]** of Seseck et al. fails to disclose **searching** a cache of reusable underlays **for underlays having the reusable document components needed by the page** because Seseck et al. teaches the comparing of a web page to a print-ready document list to determine whether a print-ready version of the web page is already stored in cache, not searching a cache of reusable underlays for underlays having the reusable document components needed by the page.

Therefore, contrary to the Examiner's allegations, Seseck et al. fails to anticipate the presently claimed invention as set by independent claim 2.

Independent Claim 8

In formulating the rejection of independent claim 8 under 35 U.S.C. §102(e), the Examiner alleges that Seseck et al. discloses, at paragraph **[0030]**, an apparatus for processing documents each represented by a document description encoded in a page description language supportive of reusable data. Moreover, the Examiner alleges that Seseck et al. discloses, at paragraphs **[0034]** and **[0072]**, a page description language interpreter that receives the document description and parses the document description into reusable document components and which combines said components into composites of reusable components and reusable underlays; at paragraph **[0039]**, an imager, communicating with the interpreter, that creates image representations of received document components; and, at paragraph **[0047]**, a reusable document component repository that stores image representations derived from a plurality of processed documents, the reusable document component repository communicating with the interpreter and the imager to supply those ones of the image representations corresponding to selected document components of the processed documents and to receive selected image representations created by the imager during the processing of documents. Based upon these allegations, the Examiner concludes that the presently claimed invention is anticipated by Seseck et al. These allegations and conclusion are respectfully traversed.

As set forth above, independent claim 8 recites an apparatus for processing documents each represented by a document description encoded in a page description language supportive of reusable data. The apparatus includes a page description language interpreter that receives the document description and parses the document description into reusable document components and which combines said components into composites of reusable components and reusable underlays; an imager, communicating with the interpreter, that creates image representations of received document components; and a reusable document component repository that stores image representations derived from a plurality of processed documents, the reusable document component repository communicating with the interpreter and the imager to supply those ones of the image representations corresponding to selected document components of the processed documents and to receive selected image representations created by the imager during the processing of documents.

For example, independent claim 8 explicitly recites a page description language interpreter that receives the document description and parses the document description into reusable document components and which combines said components into composites of reusable components and reusable underlays. In contrast, Seseck et al. teaches the retrieving of a web page data and comparing it with the print-ready documents in the printer cache.

More specifically, Seseck et al. discloses, at paragraph **[0034]**:

> The web page data is received by browsing module 158 and stored in memory 150. Some of the data received as part of a web page may include a timestamp indicating the last time the web page was modified (the timestamp may be part of the page that is displayed by a browser when displaying the page, or alternatively included in control information that is not displayed). Controller 156 compares the web page to print-ready document list 164 to determine whether a print-ready version of this web page is already stored in cache 166. This comparison can be performed in a variety of different manners, and in one implementation list 164 includes, for each file in cache 166, the URL of the web page corresponding to the file. The URL received from client 134 can then be compared to the URLs in list 164, and if the received URL exists in list 164 then a print-ready version of the document is already in cache 166. Alternatively, the determination of whether a print-ready version of a web page is already stored in cache 166 may be performed after the indication of the web page is received from client 134, but prior to retrieval of the web page from server 136 by browsing module 158. However, even if the web page does already exist in cache 166, at least a portion of the web page is still retrieved from server 136 so that the timestamps can be compared, as discussed below.

Also, Seseck et al. discloses, at paragraph **[0072]**:

> Initially, a print request is received (act 302) and the cache checked to determine whether the document for which printing is requested is in the cache (act 304). If the document is not in the cache, then the document is obtained form a remote source (e.g., the client requesting printing of the document) for printing (act 306), and printed (act 314). However, if the document is in the cache, then a check is made as to whether the document has dynamic portions (act 308). This check can be performed in a variety of different manners, such as based on whether the URL indicates a web page written in as a Java Server Page or Active Server Page. If the document does not have dynamic portions then the document is obtained from the cache for printing (act 310), and printed

> (act 314). However, if the document does have dynamic portions, then the static portion(s) of the document are obtained from the cache and the dynamic portion(s) are obtained from a remote source for printing (act 312), and printed (act 314).

As disclosed by Seseck et al., paragraphs **[0034]** and **[0072]** of Seseck et al. disclose that the comparing of the web page received from a server with the print-ready documents stored in a print-ready cache. Moreover, paragraphs **[0034]** and **[0072]** of Seseck et al. disclose that a static portion (reuseable portion) and a dynamic portion (non-reuseable portion) are obtained for printing.

On the other hand, independent claim 8 recites a page description language interpreter that **receives the document description and parses the document description into reusable document components** and **which combines said components into composites of reusable components and reusable underlays**. The teaching by Seseck et al. of combining a static portion (reuseable portion) and a dynamic portion (non-reuseable portion) fails to anticipate **combining parsed reusable document components into composites of reusable components and reusable underlays**.

Therefore, contrary to the Examiner's allegations, Seseck et al. fails to anticipate the presently claimed invention as set by independent claim 8.

Independent Claim 17

In formulating the rejection of independent claim 17 under 35 U.S.C. §102(e), the Examiner alleges that Seseck et al. discloses, at paragraph **[0034]**, receiving a document description including at least one selected reusable document component and combining said components into composites of reusable components and reusable underlays; at paragraph **[0035]**, querying a reusable document component repository containing stored image representations of reusable document components to locate a selected stored image representation corresponding to the selected reusable document component; at paragraphs **[0036]**, conditional upon the querying, identifying one of the stored image representations as corresponding to the selected reusable document component and retrieving the selected stored image representation corresponding to the selected reusable document component, or, not identifying one of the stored image representations as corresponding to the selected reusable document component, generating an image representation for the selected reusable document component, and storing the generated image representation in the reusable document component repository; and, at paragraph **[0037]**, converting the document description to a document image representation, the converting including incorporating the selected or generated image representation corresponding to the selected reusable document into the document image representation. Based upon these allegations, the Examiner concludes that the presently claimed invention is anticipated by <u>Seseck et al</u>. These allegations and conclusion are respectfully traversed.

As set forth above, independent claim 17 recites a document construction method by receiving a document description including at least one selected reusable document component and combining said components into composites of reusable components and reusable underlays; querying a reusable document component repository containing stored image representations of reusable document components to locate a selected stored image representation corresponding to the selected reusable document component; conditional upon the querying, identifying one of the stored image representations as corresponding to the selected reusable document component and retrieving the selected stored image representation corresponding to the selected reusable document component, or, not identifying one of the stored image representations as corresponding to the selected reusable document component, generating an image representation for the selected reusable document component, and storing the generated image representation in the reusable document component repository; and converting the document description to a document image representation, the converting including incorporating the selected or generated image representation corresponding to the selected reusable document into the document image representation.

For example, independent claim 17 explicitly recites receiving a document description including at least one selected reusable document component and combining said components into composites of reusable components and reusable underlays. In contrast, Seseck et al. teaches the retrieving of a web page data and comparing it with the print-ready documents in the printer cache.

More specifically, Seseck et al. discloses, at paragraph **[0034]**:

> The web page data is received by browsing module 158 and stored in memory 150. Some of the data received as part of a web page may include a timestamp indicating the last time the web page was modified (the timestamp may be part of the page that is displayed by a browser when displaying the page, or alternatively included in control information that is not displayed). Controller 156 compares the web page to print-ready document list 164 to determine whether a print-ready version of this web page is already stored in cache 166. This comparison can be performed in a variety of different manners, and in one implementation list 164 includes, for each file in cache 166, the URL of the web page corresponding to the file. The URL received from client 134 can then be compared to the URLs in list 164, and if the received URL exists in list 164 then a print-ready version of the document is already in cache 166. Alternatively, the determination of whether a print-ready version of a web page is already stored in cache 166 may be performed after the indication of the web page is received from client 134, but prior to retrieval of the web page from server 136 by browsing module 158. However, even if the web page does already exist in cache 166, at least a portion of the web page is still retrieved from server 136 so that the timestamps can be compared, as discussed below.

Also, Seseck et al. discloses, at paragraph **[0072]**:

> Initially, a print request is received (act 302) and the cache checked to determine whether the document for which printing is requested is in the cache (act 304). If the document is not in the cache, then the document is obtained form a remote source (e.g., the client requesting printing of the document) for printing (act 306), and printed (act 314). However, if the document is in the cache, then a check is made as to whether the document has dynamic portions (act 308). This check can be performed in a variety of different manners, such as based on whether the URL indicates a web page written in as a Java Server Page or Active Server Page. If the document does not have dynamic portions then the document is obtained from the cache for printing (act 310), and printed (act 314). However, if the document does have dynamic portions, then the static portion(s) of the document are obtained from the cache and the

dynamic portion(s) are obtained from a remote source for printing (act 312), and printed (act 314).

As disclosed by Seseck et al., paragraphs **[0034]** and **[0072]** of Seseck et al. disclose that the comparing of the web page received from a server with the print-ready documents stored in a print-ready cache. Moreover, paragraphs **[0034]** and **[0072]** of Seseck et al. disclose that a static portion (reuseable portion) and a dynamic portion (non-reuseable portion) are obtained for printing.

On the other hand, independent claim 17 recites receiving a document description including at least one selected reusable document component and **combining the components into composites of reusable components and reusable underlays**. The teaching by Seseck et al. of combining a static portion (reuseable portion) and a dynamic portion (non-reuseable portion) fails to anticipate **combining the components into composites of reusable components and reusable underlays**.

Therefore, contrary to the Examiner's allegations, Seseck et al. fails to anticipate the presently claimed invention as set by independent claim 17.

6. Dependent Claims

With respect to dependent claims 9-16 and 18-29, the Applicants, for the sake of brevity, will not address the reasons supporting patentability for these individual dependent claims, as these claims depend directly or indirectly from allowable independent claims 8 and 17. The Applicants reserve the right to address the patentability of these dependent claims at a later time, should it be necessary.

Accordingly, in view of the remarks set forth above, the Examiner is respectfully requested to reconsider and withdraw the rejection under 35 U.S.C. §102(e).

CONCLUSION

Accordingly, in view of all the reasons set forth above, the Examiner is respectfully requested to reconsider and withdraw the present rejections. Also, an early indication of allowability is earnestly solicited.

Respectfully submitted,

Michael J. Nickerson
Registration Number: 33,265
Basch & Nickerson LLP
1777 Penfield Road
Penfield, New York 14526
Telephone: (585) 899-3970
Customer No. 75931

MJN/mjn